# CHESS DIGITAL CLOCK

ROSMIRA BINTI ROSLAN

UNIVERSITI MALAYSIA PAHANG

"I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the Bachelor Degree of Electrical Engineering (Electronics)"

Signature : _____

Name : NURUL HAZLINA BINTI NOORDIN

Date : 12 NOVEMBER 2008

# CHESS DIGITAL CLOCK

ROSMIRA BINTI ROSLAN

This thesis is submitted as partial fulfillment
of the requirements for the award of the Bachelor of
Electrical Engineering (Hons.) (Electronics)

Faculty of Electrical & Electronics Engineering
Universiti Malaysia Pahang

NOVEMBER 2008

**Source code for Menu**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity first is
port (  clk, reset : in std_logic;
        swA : in std_logic;
         swB : in std_logic;
         swC : in std_logic;
         outA : out std_logic;
        outB : out std_logic;
        outC : out std_logic      );
end first;
architecture Behavioral of first is
begin
process (swA,swB,swC,reset,clk) begin

if reset = '1' then outA <= '0'; outB <= '0' ; outC <= '0';
else if (clk'event and clk='1') then
          if swA = '1' then outA <= '1'; outB <= '0' ; outC <= '0';
          else if swB = '1' then outB <= '1'; outA <= '0' ; outC <= '0';
          else if swC = '1' then outC <= '1'; outA <= '0' ; outB <= '0';
end if;
end if;
end if;
end if;
end if;
end process;
end Behavioral;
```

**Source code for Blitz**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity b is
Port(  clk, reset          : in std_logic;
       count      : in std_logic;
       s_2, m_2   : buffer integer range 0 to 60;
        j_2        : buffer integer range 0 to 10          );
end b;
architecture Behavioral of b is
signal temp_s2, temp_m2 : integer range 0 to 60;
signal temp_j2              : integer range 0 to 10;
begin
process ( clk,reset) begin
       if reset = '1' then temp_s2 <= 0;
       else if (clk'event and clk='1') then
              if count = '1' then temp_s2<= temp_s2-1;
              if temp_s2 = 0 then temp_s2 <=59; temp_m2<= temp_m2-1;
              if temp_m2 =0 then temp_m2 <=59; temp_j2<= temp_j2-1;
end if;
end if;
end if;
end if;
end if;
end process;
       s_2 <= temp_s2 ;
       m_2 <= temp_m2 ;
       j_2 <= temp_j2 ;
end Behavioral;
```

**Source code for Standard (1)**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity se is
Port(  clk, reset            : in std_logic;
          count              : in std_logic;
          infopponent        : in std_logic;
          s_3, m_3           : buffer integer range 0 to 60 ;
           j_3               : buffer integer range 0 to 10            );
end se;
architecture Behavioral of se is
signal temp_s3, temp_m3  : integer range 0 to 60  ;
signal temp_j3 : integer range 0 to 10;
begin
process ( clk,reset) begin
          if reset = '1' then
          temp_s3 <= 0;
          temp_m3 <= 30;
          temp_j3 <= 1;
          else if (clk'event and clk='1') then
          if count = '1' then temp_s3<= temp_s3-1;
          if infopponent = '1' then temp_s3 <= temp_s3+30;
          else if temp_s3 = 0 then temp_s3 <= 59; temp_m3 <= temp_m3-1;
          else if temp_m3 = 0 then temp_m3 <= 59; temp_j3 <= temp_j3-1;
          else if temp_j3 = 0 then temp_j3 <= 9;
          else if temp_s3 > 59 then temp_s3 <= temp_s3-60; temp_m3 <= temp_m3
+1;
          else if temp_m3 > 59 then temp_m3 <= temp_m3-60; temp_j3 <= temp_j3
+1;
end if;
end if;
```

```
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end process;
        s_3 <= temp_s3 ;
         m_3 <= temp_m3 ;
        j_3 <= temp_j3 ;
end Behavioral;
```

**Source code for Standard (2)**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity sg is
Port(  clk, reset            : in std_logic;
       count                 : in std_logic;
        infopponentg  : in std_logic;
       trigger               : buffer integer range 0 to 41;
       s_4, m_4          : buffer integer range 0 to 60 ;
        j_4                  : buffer integer range 0 to 10 );
end sg;
architecture Behavioral of sg is
signal temp_s4, temp_m4  : integer range 0 to 60  ;
signal temp_j4 : integer range 0 to 10;
signal temp_trigger : integer range 0 to 41;
begin
process ( clk,reset) begin
       if reset = '1' then
       temp_s4 <= 0;
       temp_m4 <= 30;
       temp_j4 <= 1;
       else if (clk'event and clk='1') then
              if count = '1' then temp_s4<= temp_s4-1;
              if infopponentg = '1' then temp_s4 <= temp_s4+30;

temp_trigger <= temp_trigger + 1;
       else if temp_trigger = 40 then temp_m4 <= temp_m4 + 40;
       else if temp_s4 = 0 then temp_s4 <= 59; temp_m4 <= temp_m4-1;
       else if temp_m4 = 0 then temp_m4 <= 59; temp_j4 <= temp_j4-1;
       else if temp_j4 = 0 then temp_j4 <= 9;
```

```
        else if temp_s4 > 59 then temp_s4 <= temp_s4-60; temp_m4 <= temp_m4
+1;
                        else if temp_m4 > 59 then temp_m4 <= temp_m4-60; temp_j4
<= temp_j4 +1;
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end if;
end process;
        s_4 <= temp_s4 ;
         m_4 <= temp_m4 ;
        j_4 <= temp_j4 ;
        trigger <= temp_trigger ;
end Behavioral;
```

**Source code for port map**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity standard is
port ( clk_T, reset_T  : in std_logic;
          swA_T          : in std_logic;
          swB_T          : in std_logic;
          swC_T          : in std_logic;
          info_T         : in std_logic;
          infog_T        : in std_logic;
          trigger_T      : buffer integer range 0 to 41;
          s_2T, m_2T     : buffer integer range 0 to 60;
           j_2T          : buffer integer range 0 to 10;
           s_3T, m_3T    : buffer integer range 0 to 60;
           j_3T          : buffer integer range 0 to 10;
          s_4T, m_4T     : buffer integer range 0 to 60;
           j_4T          : buffer integer range 0 to 10);
end standard;
architecture structural of standard is
component first
port (   clk, reset : in std_logic;
          swA : in std_logic;
         swB : in std_logic;
         swC : in std_logic;
         outA : out std_logic;
        outB : out std_logic;
        outC : out std_logic              );
end component;
component b
port ( clk, reset : in std_logic;
          count      : in std_logic;
```

```
        s_2, m_2       : buffer integer range 0 to 60;
         j_2          : buffer integer range 0 to 10);
end component;
component se
port ( clk, reset : in std_logic;
        count      : in std_logic;
        infopponent : in std_logic;
         s_3, m_3       : buffer integer range 0 to 60;
         j_3         : buffer integer range 0 to 10);
end component;
component sg
port ( clk, reset : in std_logic;
        count      : in std_logic;
        infopponentg: in std_logic;
        trigger    : buffer integer range 0 to 41;
        s_4, m_4       : buffer integer range 0 to 60;
         j_4         : buffer integer range 0 to 10);
end component;
signal bus_1: std_logic;
signal bus_2 :std_logic;
signal bus_3 :std_logic;


begin
U1 : first port map ( clk=>clk_T,
                      reset=>reset_T,
                      swA=>swA_T,
                      swB=>swB_T,
                       swC=>swC_T,
                      outA=>bus_1,
                      outB => bus_2,
                      outC => bus_3);
U2 : b port map ( clk=>clk_T,
                  reset=>reset_T,
```

```
                count=>bus_1,
                s_2=>s_2T,
                m_2=>m_2T,
                j_2=>j_2T );
U3 : se port map ( clk=>clk_T,
                reset=>reset_T,
                count=>bus_2,
                infopponent=>info_T,
                s_3=>s_3T,
                 m_3=>m_3T,
                 j_3=>j_3T);
U4 : sg port map ( clk=>clk_T,
                reset=>reset_T,
                count=>bus_3,
                infopponentg=>infog_T,
                 trigger=>trigger_T,
                s_4=>s_4T,
                m_4=>m_4T,
                j_4=>j_4T );
end structural;
```

# CHAPTER 1

# INTRODUCTION

## 1.1 Background

This chapter focuses on the methodologies for the development and implementation of the Chess Digital Clock project. The project includes the ISE software coding (ISE Design Suite 10.1) and FPGA Xilinx board.

In very general terms, coding can be said that the communication between human and technologies. One example of an application is a Chess Digital Clock. This application (software coding) executes on FPGA board (technology) that can support that application.

This chapter also explains the overview of Chess Digital Clock project, the objectives of the project, project scopes and thesis outline.

## 1.2 Overview of Chess Digital Clock

Sport is an activity that is governed by a set of rules or customs and often engaged in competitively. Sports commonly refer to activities where the physical capabilities of the competitor are the sole or primary determiner of the outcome (winning or losing), but the term is also used to include activities such as mind sports (a common name for some card games and board games with little to no element of chance) and motor sports where mental acuity or equipment quality are major factors. [6]

Besides casual games without exact timing, chess is also played with a time control, mostly by club and professional players. If a player's time runs out before the game is completed, the game is automatically lost. The timing ranges from long games played up to seven hours to shorter rapid chess games lasting usually 30 minutes or one hour per game. Even shorter is blitz chess with a time control of three to fifteen minutes for each player and bullet chess (under three minutes). [7]

The development of this Chess Digital Clock consists of two parts. Project part one which is concentrate on software coding. The software that is used in this project is ISE software in VHDL code. The software coding started with ISE 6.0 and its simulation done with MXE (ModelSim Xilinx Edition). After the several months the using software of ISE 6.0 changes to the latest version which is ISE 10.0. This latest version of ISE software is much easier in the simulation. ISE Design Suite 10.1 includes the Integrated Software Environment (ISE), ChipScope Pro, Xilinx Embedded Development Kit (EDK), DSP Tools (including AccelDSP and System Generator), and Plan Ahead/PlanAhead Lite. It also describes how to use Xilinx online documentation.

The second part of the project is concentrate in implementing the software to the FPGA board. FPGA board that used in the project is FPGA Xilinx board. FPGA requires user hardware programming to perform the desired operation. Xilinx Spartan FPGAs are ideal for low-cost, high-volume applications and are targeted as replacements for fixed-logic gate arrays and ASSP products such as bus interface chip sets.

Figure 1.1 shows the methodology of the Chess Digital Clock. The first step is to design the digital concept. The designer should familiar with the module which has to be design. This is the part of starting the coding. The design entry where is the design is created and entered into the computer in the form of an HDL source code, using a design entry tool. After all modules have been completely designed, the final design is portmapping. Port-map is the place where is the combination of all modules. It is a method for associating signals with their respective ports.

After a design is generated, the resulting VHDL code may be simulated for the behavior of the designed circuit using a VHDL (very high-speed integrated circuit hardware description language) simulation tool. The VHDL code generated from design entry tool is passed to the synthesis module, converting the code to a logic netlist file.

The netlist obtained from the synthesis tool may be verified for design correctness using a functional simulation tool. The netlist file is converted to a physical design in the target implementation technology. Where each logic function is mapped (implemented) to the logic elements available in the target chip.

The physical layout obtained from implementation process can be simulated to verify the design, but this time, with timing information.

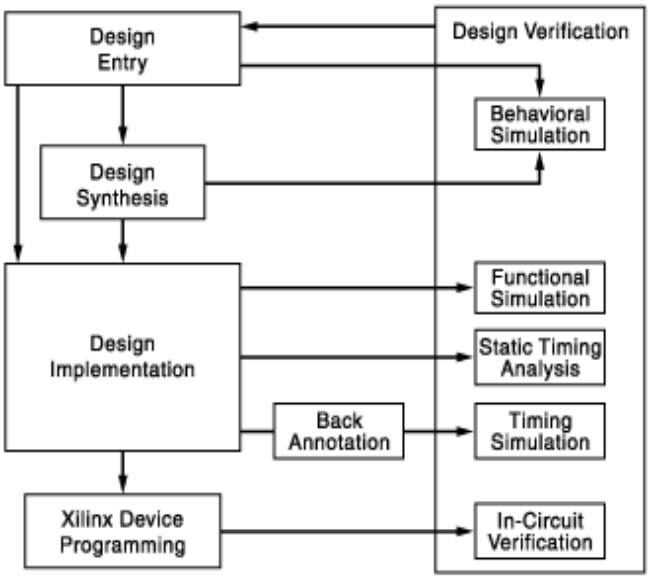


Figure 1.1 : General Design Flow

### 1.3 Project Objectives

The overall target of the whole project is to allow people playing chess interestingly. However the objectives of the project are display the three time controls in numbers and implement the design onto FPGA board.

### 1.4 Project Scopes

This project is concentrates on the time of playing chess according to the type of playing through the display Chess Digital Clock. The ISE 10.0 software is used to construct the project that contain three different setting time of playing chess. The successful in simulation of the design is one of the scopes of the project. To achieve the objective of the project, the three different setting times of playing chess need to

implement in the FPGA Xilinx board. The product of the project help the user of the digital clock go easier while playing chess.

## 1.5 Thesis Outline

Chapter 1 focuses on the methodologies for the development and implementations of the user of the Chess Digital Clock. It gives a brief the steps and the purpose of the Chess Digital Clock.

Chapter 2 explains the background of the ISE software and FPGA Xilinx board and the relationship of each part in develop Chess Digital Clock. The concept of the software and the FPGA board are the two essentials concept as a guide to the construction of Chess Digital Clock. In this chapter also explain the three rules of playing chess for designing the system.

Chapters 3 explain and discuss the process of using and control the chess clock. It discusses the brief review of how the Chess Digital Clock works and the algorithm of the user.

Chapter 4 discusses all the result obtained and the limitation of the project. All discussions are concentrating on the result and the overall performance of the project

Chapter 5 discusses the conclusion of development of whole the system. This chapter also discusses the problem and the recommendation for this project and overall of the system for the future development or modification. Besides that, this chapter also explains about the costing and commercialization.

# CHAPTER 2

# BACKGROUND

## 2.1 Background

This chapter explains the background of ISE Software and FPGA Xilinx board and the relationship of each part in develop Chess Digital Clock. These are the main tools as a guide to the development of the Chess Digital Clock. Else in this chapter, also explain about the three of the chess time control game. These rules of the chess game must be considered to design the concept of the Chess Digital Clock.

## 2.2 ISE software

WebPack is a shell script for automatically packing Web sites by shrinking them without affecting their functionality or appearance. It is also useful for lossless shrinking image collections and locating corrupt files. It works by stripping unnecessary information and optimizing the compression of images, and by removing comments/whitespace from HTML, using readily-available tools. [1]

A Webpack is a packaged service to make top quality web sites accessible to small businesses at minimal cost. A Webpack sites contain everything that a small business requires to project a professional image online. Individual Webpack ISE

modules give you the ability to tailor the design environment to your chosen PLDs as the preferred design flow. In general, the design flow for FPGAs and CPLDs is identical. You can choose wether to enter the design in schematic form or in HDL, such as VHDL, Verilog or ABEL. The design can also comprise of a mixture of schematic diagrams and embedded HDL symbols. There is also a facility to create state machines in a diagrammatic form and let the software tools generate optimized code from a state diagram. WebPACK ISE software incorporates a Xilinx version of the ModelSim simulator from Model Technology (a Mentor Graphics company), referred to as MXE (ModelSim Xilinx Edition). This powerful simulator is capable of simulating functional VHDL before synthesis, or simulating after the implementation process of timing verification. WebPACK ISE software offers an easy-to-use GUI to visually create a test pattern. A testbench is then generated and compiled into MXE, along with the design under test. The flow diagram below shows the similarities and differences between CPLD and FPGA software flows. [2]

Xilinx programmable logic solutions help minimize risks for electronic equipment manufacturers by shortening the time required to develop products. Webpack ISE design software offers a complete design suite based on the Xilinx ISE series software.

### 2.2.1 ISE Design Suite 10.0

For this project, the design is enter in HDL (Hardware Description Language), which is VHDL code.

VHDL is an industry standard language for modeling digital circuits. It is intended for design documentation and simulation. The basics of VHDL are including VHDL Design Unit, VHDL Data Object and Types, and VHDL operators.

A typical VHDL module consists of library declarations, an entity, and architecture. The library declarations are needed to tell the compiler which packages are required.

Webpack ISE software incorporates a Xilinx version of the ModelSim simulator from Model Technology (a Mentor Graphics company), referred to as MXE (ModelSim Xilinx Edition). This powerful simulator is capable of simulating functional VHDL before synthesis, or simulating after the implementation process for timing verification.

Individual can design and verify the unique circuits in Xilinx programmable devices much faster than by choosing traditional methods such as mask-programmed, fixed logic gate arrays.

## 2.3 Field Programmable Gate Arrays (FPGAs)

Short for **F**ield-**P**rogrammable **G**ate **A**rray, a type of logic chip that can be programmed. An FPGA is similar to a PLD, but whereas PLDs are generally limited to hundreds of gates, FPGAs support thousands of gates. They are especially popular for prototyping integrated circuit designs. Once the design is set, hardwired chips are produced for faster performance. [3]

A field-programmable gate array is a semiconductor device containing programmable logic components called logic blocks, and programmable interconnects. Logic blocks can be programmed to perform the function of basic logic gates such as AND, and XOR, or more complex combinational functions such as decoders or mathematical functions. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

Field-Programmable Gate Arrays (FPGAs) have become one of the key digital circuit implementation media over the last decade. A crucial part of their creation lies in their architecture, which governs the nature of their programmable logic functionality and their programmable interconnect. FPGA architecture has a dramatic effect on the quality of the final device's speed performance, area efficiency, and power consumption.

### 2.3.1 FPGA Xilinx board

The ISE™ design flow comprises the following steps: design entry, design synthesis, design implementation, and Xilinx® device programming. Design verification, which includes both functional verification and timing verification, takes places at different points during the design flow. [5]

In this project, using FPGA Xilinx board is the best choose. This is the latest upgrade product of Xilinx. Xilinx is also actively developing breakthrough technology that will enable the hardware in Xilinx-based systems to be upgraded remotely over any kind of network – including the Internet – even after the equipment has been shipped to a customer.

Xilinx Spartan FPGAs are ideal for low-cost, high-volume applications and are targeted as replacements for fixed-logic gate arrays and ASSP (Application-Specific Standard product) products such as bus interface chip sets.

## 2.4 Time Control

Chess clock are really two connected clocks. While player A is thinking, his clock is running and the player B clock is stopped. Once the player A makes a move and he hit the clock, which stop his clock and starts the player B clock. There is only one clock running at a time because, each player gets their own separate amount of time. This is to allow for the fact that some moves take only a few seconds to play, while others might take several minutes, depending on the complexity of the position.

The term time control refers to the amount of time each player has to make some or all moves during a game. The three different kinds of time control are Blitz, Standard (1) and Standard (2). The different names distinguish the different maximum duration of a game.

### 2.4.1 Blitz

In Blitz chess, each player gets a fixed amount of time for the entire game. For example of Blitz kind of time control is five minutes per side game. Each player gets five minutes on their clock, so the time might be set to 4:55 on each side.

As the player with White is thinking about the first move, his clock is running. After a few seconds, he makes the move and hits his clock. This starts his

opponent's clock. He can take as much time as he wants to for each move. Then he hits his clock. This goes on, back and forth.

In Blitz, when the five minutes is up, the person’s clock is the first one out of time, he lose, regardless of the position on the board unless his opponent has insufficient material to mate. Blitz chess is very exciting, and lots of fun for social games and one-day tournaments.

### 2.4.2 Standard (1)

Most serious international tournaments, and many amateur tournaments, use a Standard (1) (quota system) for time controls. As in Blitz, each player gets their own time, and need to finish their game in the time that allocated. The difference is that the player will be given more time to continue playing. That is mean some extra time will be added to their time when the game is running.

Each player also gets a fixed amount of time for the entire game. But when the players hit his clock after make their move, then their time will get 30 seconds extra time to they continue the game. Every time when they hit their clock, the will be added 30 seconds at their clock. This make the game held longer than Blitz. The rest of the game runs the same with the Blitz method.